Linear Network Codes and Systems of Polynomial Equations

Randall Dougherty, Chris Freiling, and Kenneth Zeger, *Fellow, IEEE*

***Abstract*—If β and γ are nonnegative integers and F is a field, then a polynomial collection $\{p_1, \ldots, p_\beta\} \subseteq \boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$ is said to be *solvable over* F if there exist $\omega_1, \ldots, \omega_\gamma \in F$ such that for all $i = 1, \ldots, \beta$ we have $p_i(\omega_1, \ldots, \omega_\gamma) = 0$. We say that a network and a polynomial collection are *solvably equivalent* if for each field F the network has a scalar-linear solution over F if and only if the polynomial collection is solvable over F. Koetter and Médard's work implies that for any directed acyclic network, there exists a solvably equivalent polynomial collection. We provide the converse result, namely, that for any polynomial collection there exists a solvably equivalent directed acyclic network. (Hence, the problems of network scalar-linear solvability and polynomial collection solvability have the same complexity.) The construction of the network is modeled on a matroid construction using finite projective planes, due to MacLane in 1936.**

A set Ψ of prime numbers is a *set of characteristics* of a network if for every $q \in \Psi$, the network has a scalar-linear solution over some finite field with characteristic q and does not have a scalar-linear solution over any finite field whose characteristic lies outside of Ψ. We show that a collection of primes is a set of characteristics of some network if and only if the collection is finite or co-finite.

Two networks $\mathcal{N}$ and $\mathcal{N}'$ are *ls-equivalent* if for any finite field F, $\mathcal{N}$ is scalar-linearly solvable over F if and only if $\mathcal{N}'$ is scalar-linearly solvable over F. We further show that every network is ls-equivalent to a multiple-unicast matroidal network.

***Index Terms*—Flow, information theory, matroids, network coding, polynomial equations.**

I. Introduction

In this paper, we first demonstrate a certain equivalence between networks and collections of polynomials. Specifically, we show that associated with every finite collection of polynomials with integer coefficients is a corresponding network which is scalar-linearly solvable precisely over those finite fields where the polynomials have a common root. A consequence is that the complexity of determining whether networks are scalar-linearly solvable over particular finite fields is equivalent to the complexity of determining whether collections of polynomial have common roots over the corresponding fields. Second, we show that the collections of prime numbers corresponding to the field characteristics of scalar-linearly solvable network alphabets are precisely those which are finite or co-finite. Finally, we show that for every network, there exists a multiple-unicast network which is matroidal (i.e., obtained from a certain matroid-to-network construction), such that the two networks are scalar-linearly solvable over exactly the same finite fields. A *network* is a finite, directed, acyclic multigraph with node set ν and edge set ϵ, together with a finite set μ called the *message set*, a *source mapping*

$$S : \nu \to 2^\mu,$$

and a *receiver mapping*

$$R : \nu \to 2^\mu$$

where 2^μ is the power set of μ. For every node u, if $S(u)$ is nonempty, then u is called a *source*, and if $R(u)$ is nonempty, then u is called a *receiver*. The elements of $S(u)$ are called the *messages generated by* u and the elements of $R(u)$ are called the *messages demanded by* u.

An *alphabet* is a set $\mathcal{A}$ with at least two elements.

For each network node u, let $\mathrm{In}(u)$ (the *inputs*) denote the union of the set of messages generated by u with the set of in-edges of u, and let $\mathrm{Out}(u)$ (the *outputs*) denote the union of the set of messages demanded by u with the set of out-edges of u. For every node u, fix an ordering of $\mathrm{In}(u)$ such that all messages in the resulting list occur before the edges in the list; the resulting ordered list is called the *input list* of u. For every edge $e = (u, v)$, an *edge function* is a map

$$f_e : \mathcal{A}^\sigma \times \mathcal{A}^\tau \to \mathcal{A}$$

where σ and τ are the number of messages and edges, respectively, in the input list of u (note that σ and τ are functions of u). For every $u \in \nu$ and $m \in R(u)$, a *decoding function* is a map

$$f_{u,m} : \mathcal{A}^\sigma \times \mathcal{A}^\tau \to \mathcal{A}$$

where σ and τ are the number of messages and edges, respectively, in the input list of u.

Given an alphabet $\mathcal{A}$, a *code* for a network is an assignment of edge functions to the network's edges and an assignment of decoding functions to the network's receiver/demanded-message pairs. If the edge functions and decoding functions in a code are linear maps with respect to a field alphabet $\mathcal{A}$, then the code is called a *scalar-linear code*.

A *message assignment* is a map $\mathcal{V} : \mu \to \mathcal{A}$. For any code and for any message assignment, we recursively extend $\mathcal{V}$ to a map $\mathcal{V} : \mu \cup \epsilon \to \mathcal{A}$ as follows. For every edge $e = (u, v)$, let

$$\mathcal{V}(e) = f_e(\mathcal{V}(u_1), \ldots, \mathcal{V}(u_\sigma), \mathcal{V}(u_{\sigma+1}), \ldots, \mathcal{V}(u_{\sigma+\tau}))$$

Manuscript received March 15, 2007; revised November 7, 2007. This work was supported by the Air Force Office of Scientific Research under Contract FA9550-06-1-0210, the Institute for Defense Analyses, the National Science Foundation, and the University of California, San Diego Center for Wireless Communications.

R. Dougherty is with the Center for Communications Research, San Diego, CA 92121-1969 USA (e-mail: rdough@ccrwest.org).

C. Freiling is with the Department of Mathematics, California State University, San Bernardino, San Bernardino, CA 92407-2397 USA (e-mail: cfreilin@csusb.edu).

K. Zeger is with the Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093-0407 USA (e-mail: zeger@ucsd.edu).

Communicated by E. Modiano, Associate Editor for Communication Networks.

Digital Object Identifier 10.1109/TIT.2008.920209

where $u_1, \ldots, u_\sigma$ are the messages generated by u and $u_{\sigma+1}, \ldots, u_{\sigma+\tau}$ are the in-edges of u. We further extend the domain of $\mathcal{V}$ to include nodes u with $|\mathrm{In}(u)| = 1$, by defining $\mathcal{V}(u) = \mathcal{V}(e)$ if u's only input is e. If x is a node with in-degree one or an edge, we say that x *carries* the symbol $\mathcal{V}(x)$. When a node u has exactly one in-edge e, we will assume without loss of generality that every out-edge of u carries the same symbol as e.

For a given network, a given code, receiver u, and message m demanded by u, if for every message assignment $\mathcal{V} : \mu \to \mathcal{A}$ we have

$$f_{u,m}(\mathcal{V}(u_1), \ldots, \mathcal{V}(u_\sigma), \mathcal{V}(u_{\sigma+1}), \ldots, \mathcal{V}(u_{\sigma+\tau})) = \mathcal{V}(m),$$

then we say that *u's demand m is satisfied*. In other words, the receiver u can recover an arbitrary instance of the message m generated by its source. A code is said to be a *solution* (over the alphabet $\mathcal{A}$) if every demand of every receiver is satisfied. If a scalar-linear code over a field alphabet $\mathcal{A}$ is a solution, then the code is called a *scalar-linear solution* over $\mathcal{A}$.

A network is said to be *solvable* if it has a solution over some finite alphabet and is *scalar-linearly solvable* if it has a scalar-linear solution over some finite field alphabet.

Throughout the remainder of the paper, for brevity, we will drop the $\mathcal{V}$ notation and simply refer to $\mathcal{V}(x)$ as x. For example, if a and b are network messages, and e is an edge satisfying $\mathcal{V}(e) = \mathcal{V}(a) + 2\mathcal{V}(b)$, then we would instead write $e = a + 2b$.

Informally, a network coding solution allows each receiver to deduce its demanded messages from its in-edges and source messages by having information propagate from the sources through the network. Each edge is allowed to be used at most once (i.e., at most one symbol can travel across each edge). For introductory material on network coding, see [11], [25].

There has been significant interest in determining the solvability, scalar-linear solvability, and vector-linear solvability of an arbitrary network with respect to a chosen alphabet. For the special case of multicast networks, Li, Young, and Cai [16] showed that if the network is solvable, then it has a scalar-linear solution over all sufficiently large finite field alphabets. Jaggi *et al.* [14] and Ho *et al.* [12] showed that every solvable multicast network with at least two receiver nodes has a linear solution with some finite field alphabet of size at most equal to the number of receiver nodes. Feder, Ron, and Tavory [10] showed that to achieve a linear solution, some solvable multicast networks asymptotically require finite field alphabets to be at least as large as twice the square root of the number of receiver nodes. Rasala Lehman and Lehman [21] constructed a similar multicast network as in [10] (in independent work) and also achieved essentially the same square root lower bound as in [10]. The result in [21] actually shows that the square root lower bound on alphabet size applies to finding any solution, not just a linear solution. They also gave an example of a multicast network which is solvable over a ternary alphabet but which has no linear solution for alphabets of cardinality less than five. Furthermore, it was shown in [21] that the problems of determining the minimum alphabet size for both linear and nonlinear solutions to a multicast network are NP-hard. Riis [22] noted in particular that every solvable multicast network has a binary linear solution in some vector dimension.

We next summarize a number of results regarding the solvability and linear solvability of general (i.e., not necessarily multicast) networks. Riis demonstrated in [22] solvable networks which can achieve binary linear solutions only if the vector dimension grows at least linearly with the number of nodes in the network. It was shown in [5], [22] that a network might have a binary nonlinear solution, but no binary linear solution. It was shown in [5] that a network might be solvable over a certain alphabet, but not over all larger alphabets. Rasala Lehman and Lehman [21] and Médard *et al.* [19] showed that a network might be solvable over all alphabets, but not linearly solvable over any alphabet. It was shown in [6] that a network might be solvable, but not vector-linearly solvable over any vector dimension or any finite field (or more general algebraic structure) alphabet. It was shown in [8] that a network might be vector-linearly solvable only over even vector dimensions. It was shown in [7] that a network might be solvable only over alphabets with odd (or alternatively, a power of two) cardinality. It was shown in [6] that a network might be scalar-linearly solvable only over finite field alphabets with odd (or alternatively, even) characteristic. In [21], Rasala Lehman and Lehman gave solvable networks whose minimum alphabet size required for a solution could be made arbitrarily large.

For a given finite alphabet, to determine whether a network is solvable or scalar-linearly solvable, one can (in principle) perform a finite exhaustive search of all possible codes for the network. If a vector dimension is also fixed, a finite search can also establish if a network is vector-linearly solvable over that dimension. There is presently no known algorithm for determining the general solvability or vector-linear solvability of an arbitrary network. The existence of an algorithm (which is apparently not computationally efficient) to determine scalar-linear solvability of an arbitrary network follows from work in [15] by Koetter and Médard, and will be discussed in Section VII. Their technique was to construct a finite collection of polynomials from an arbitrary network, such that for each finite field, the polynomials have a common root over the field if and only if the network has a scalar-linear solution over the field.

Throughout this paper, polynomials will have integer coefficients and will use the variables $\alpha_1, \alpha_2, \ldots$. For nonnegative integers β and γ, any finite set

$$\mathcal{P} = \{p_1, \ldots, p_\beta\} \subseteq \boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$$

will be called a *polynomial collection*. If F is a field, then a polynomial collection is said to be *solvable over F* if there exist $\omega_1, \ldots, \omega_\gamma \in F$ such that for all $i = 1, \ldots, \beta$ we have

$$p_i(\omega_1, \ldots, \omega_\gamma) = 0.$$

We say that a network and a polynomial collection are *solvably equivalent* if for each field F the network has a scalar-linear solution over F if and only if the polynomial collection is solvable over F.

In this paper, we present an algorithm in Section II for constructing a network from any polynomial system. Our main results are as follows: the network is scalar-linearly solvable

over the same fields as those for which the polynomials have common roots (Theorem I.2), the constructed network is always matroidal (Theorem I.3), every network is scalar-linearly solvable on the same set of fields as a multiple-unicast matroidal network (Corollary I.8), and the collections of prime numbers corresponding to the field characteristics of scalar-linearly solvable network alphabets are characterized as either finite or co-finite (in Theorem I.9). The remainder of this section discusses the results in more detail. The proof of Theorem I.3 appears in Section IV, and the proofs of Theorem I.2 and Theorem I.9 appear in Section V.

In the construction in [15], for each out-edge of a node, the coefficients multiplying the in-edges of that node correspond to variables in the resulting polynomials. The form of their polynomials is restricted—for example, no variables can occur to a power greater than one. In contrast, in our construction the polynomials are located at only specific places in the network and are combined by different rules and any polynomial can be used.

Let Ψ be an arbitrary collection of integers of the form q^i, where q is prime and $i \geq 1$. We say that Ψ is the *solvability set* for a network (respectively, polynomial collection) if for every finite field F, the network is scalar-linearly solvable (respectively, polynomial collection is solvable) if and only if $|F| \in \Psi$. The set of primes q, such that q^i lies in the solvability set for some $i \geq 1$, is called the *set of characteristics*[1] for a network (respectively, polynomial collection). (It will be shown in Section VI, part 7, that for any prime q and any positive integer i, one can find a network such that q^i is the smallest power of q in the solvability set of the network. However, one can ask, given a network and a prime q in the set of characters of the network, whether one can give a reasonable bound on the smallest i such that q^i is in the solvability set of the network. We have no answer to this question at present, but some related results are given in the Appendix of [2].)

The following theorem leads to an algorithm (via Gröbner bases [2]) for determining whether a network has a scalar-linear solution. No such algorithm is presently known for determining whether a network has a general nonlinear solution.

Theorem I.1: (follows from Koetter–Médard [15])
Every directed acyclic network has a solvably equivalent polynomial collection.

In this paper, we provide the following converse result.

Theorem I.2: (Converse to Theorem I.1)
Any polynomial collection has a solvably equivalent directed acyclic network.

Furthermore, the solvably equivalent network in Theorem I.2 is given constructively and is matroidal, as stated in the next theorem.

Theorem I.3: If a polynomial collection $\mathcal{P}$ is solvable over some finite field, then any network constructed, as in Section II, from $\mathcal{P}$ is matroidal.

The next definition is taken from [9] (the acronym "CSLS" stands for "coding solvability, linear solvability").

[1]This terminology is taken from [1].

Definition I.4: Two networks $\mathcal{N}$ and $\mathcal{N}'$ are *CSLS-equivalent* if the following two conditions hold.

1) For any finite alphabet $\mathcal{A}$, $\mathcal{N}$ is solvable over $\mathcal{A}$ if and only if $\mathcal{N}'$ is solvable over $\mathcal{A}$.
2) For any finite field F and any positive integer k, $\mathcal{N}$ is vector-linearly solvable over F in dimension k if and only if $\mathcal{N}'$ is vector-linearly solvable over F in dimension k.

The following definition gives a type of equivalence that is weaker than CSLS (the acronym "ls" stands for "(scalar) linear solvability").

Definition I.5: Two networks $\mathcal{N}$ and $\mathcal{N}'$ are *ls-equivalent* if for any finite field F, $\mathcal{N}$ is scalar-linearly solvable over F if and only if $\mathcal{N}'$ is scalar-linearly solvable over F.

Theorem I.6: (see [9, Theorem II.1])
Any network is CSLS-equivalent to a multiple-unicast network.

The next theorem shows that if Theorem I.6 is applied to a matroidal network, then the resulting multiple-unicast network can also be taken to be matroidal.

Theorem I.7: (see [8, Corollary VII.8])
Any matroidal network is CSLS-equivalent to a multiple-unicast matroidal network.

The next corollary follows from our main result in Theorem I.2 together with several previous results. It demonstrates that, when considering which finite fields arbitrary networks are scalar-linearly solvable over, it suffices to restrict attention to the subclass of networks which are simultaneously multiple-unicast and matroidal.

Corollary I.8: Any network is ls-equivalent to a multiple-unicast matroidal network.

Proof: Let $\mathcal{N}$ be an arbitrary network. In [8, Theorems VIII.1 and VIII.2] it was shown that there exists a network $\mathcal{N}'$ (namely, a version of the "Vámos network") which is multiple-unicast and matroidal, but not scalar-linearly solvable. If $\mathcal{N}$ is not scalar-linearly solvable, then it is clearly ls-equivalent to $\mathcal{N}'$.

Now suppose $\mathcal{N}$ is scalar-linearly solvable. Then, by Theorem I.1, there exists a polynomial collection $\mathcal{P}$ which is solvably equivalent to $\mathcal{N}$. By Theorems I.2 and I.3, there exists a network $\mathcal{N}''$ which is matroidal and solvably equivalent to $\mathcal{P}$, and therefore ls-equivalent to $\mathcal{N}$. By Theorem I.7, there exists a multiple-unicast matroidal network $\mathcal{N}'''$ which is CSLS-equivalent to $\mathcal{N}''$, and thereby ls-equivalent to $\mathcal{N}$. □

Theorem I.9: A set of prime numbers is the set of characteristics of some network if and only if the set is finite or co-finite.

Theorem I.1 and our Theorem I.2 together indicate that determining the scalar-linear solvability of a directed acyclic network over a field F is computationally equivalent to determining whether a collection of polynomials has a common root over F. Given any algorithm for determining scalar-linear network solvability, our result gives an algorithm for determining polynomial solvability. This is a "many-to-one reduction" (i.e., it converts a single instance of the polynomial solvability problem to a single instance of the network scalar-linear solvability problem with the same answer). The reduction causes at

TABLE I
INSTANTIATIONS OF THE GENERIC NETWORK COMPONENT SHOWN IN FIG. 2. EACH LINE IN THE TABLE GIVES THE FIVE VALUES THAT ARE USED TO FORM A SPECIFIED COMPONENT

Component	Input_1	Input_2	Input_3	New node	New receiver's demand
$\mathcal{C}_1(i)$	x_0	x_1	z_∞	x_{α_i}	a
$\mathcal{C}_2(q)$	z_0	z_∞	x_q	z_q	c
$\mathcal{C}_3(q)$	x_q	z_1	z_0	y_q	a
$\mathcal{C}_4(q)$	x_q	z_0	z_∞	u_q	c
$\mathcal{C}_5(q)$	x_0	z_q	z_∞	u_{-q}	c
$\mathcal{C}_6(q,r)$	z_q	u_r	z_∞	x_{q+r}	a
$\mathcal{C}_7(q,r)$	z_q	y_r	z_∞	x_{qr}	a

most a linear blowup in input size, in the following sense: the number of nodes and edges in the resulting network is at most a linear function of the number of steps (variable retrievals and arithmetic operations) needed to compute the values of the polynomials in the collection. In terms of bit representations, it is at most an $O(n \ln n)$ blowup. This many-to-one reduction has the additional property that, given a scalar-linear solution to the network, we can directly reconstruct a solution to the polynomial collection (shown in part 2 of Section V).

It can be shown (via Gröbner bases) that the sets of characteristics of polynomial collections are precisely the sets of primes which are finite or co-finite (by Lemma V.1 and Examples (1a) and (1b) in Section VI). In contrast, there has been no known characterization of the sets of characteristics or the solvability sets of networks. If Ψ is the solvability set of a network and $n \in \Psi$, then $n^i \in \Psi$ for all positive integers i (by using Cartesian product codes). While there are an uncountable number of sets of powers of primes closed under exponentiation, there are only a countably infinite number of solvability sets since there are only a countable number of networks and polynomial collections.

A fundamental problem is to determine which sets of integers can be solvability sets and which can be sets of characteristics for networks. Theorem I.1 shows that every network solvability set is also a polynomial collection solvability set. Our Theorem I.2 shows that every polynomial collection solvability set is also a network solvability set. Thus, the network solvability sets are the same as the polynomial collection solvability sets. Our Theorem I.9 shows that a set of primes is the set of characteristics of a network if and only if the set of primes is finite or co-finite.

We note that throughout this paper, whenever we refer to a field without the modifier "finite," we have done so intentionally so as not to imply a restriction to finite fields in such cases.

In Section II, the formal construction of networks from polynomial collections is given, and in Section III, a concrete example of the construction is demonstrated. In Section IV, the matroidality of the constructed networks is derived in terms of a finite projective geometry. In Section V, the main theorem, about polynomial collections having solvably equivalent networks, is given. Finally, Section VI gives examples of various polynomial collections, and Section VII discusses the complexity of various network and polynomial collection problems.

II. NETWORK CONSTRUCTION FROM POLYNOMIAL SYSTEM

In this section, we present an algorithm for constructing a directed acyclic network $\mathcal{N}$ from a finite polynomial collection $\mathcal{P} = \{p_1, \ldots, p_\beta\} \subset \boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$, for $i = 1, \ldots, \beta$. The

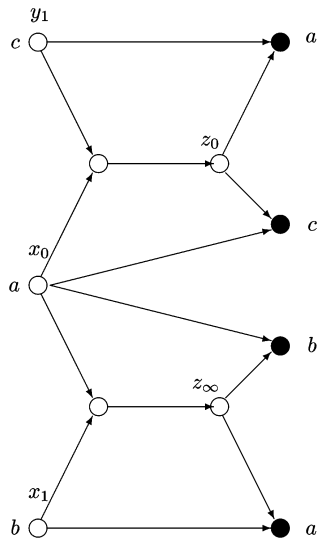

Fig. 1. Network component $\mathcal{C}_0$. The leftmost three nodes are sources, generating messages c, a, and b from top to bottom, respectively. The rightmost four nodes are receivers and demand messages a, c, b, and a, respectively. Five of the nodes are labeled by x_0, x_1, y_1, z_0, or z_∞.

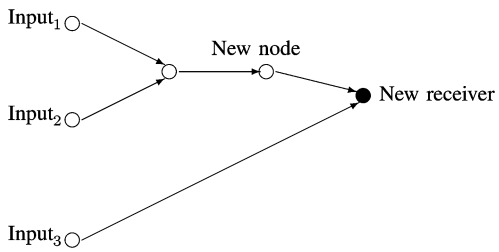

Fig. 2. A generic network component $\mathcal{C}_i$ for $1 \leq i \leq 7$. Input_1, Input_2, and Input_3 are existing nodes in the network and the remaining nodes and edges in component $\mathcal{C}_i$ are new. The rightmost node, "New receiver," demands one message. Table I lists seven different instantiations of this generic network component that are used in a network construction.

network will be built piece by piece from eight building block components, $\mathcal{C}_0, \ldots, \mathcal{C}_7$, which are shown in Figs. 1 and 2 (using Table I). The messages will be a, b, and c. Certain nodes of the network will be labeled by x_q, y_q, u_q, or z_q, where for each such node, q is some polynomial in $\boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$. For example, the sources for a, b, c will be nodes x_0, x_1, y_1, respectively. The reasoning behind the notation for the nodes will be explained later. During the construction, we will label various nodes with polynomials and will later demonstrate a connection between these polynomials and the alphabet symbols carried by these nodes. It will be demonstrated in Section V that this construction algorithm produces a network such that for any field F, the network has a scalar-linear solution over F if and only if the polynomial

collection $\mathcal{P}$ has a solution over F. The motivation for the steps in the construction will be clarified in Section IV.

The network construction process consists of the following steps.

Step (1): Start with component $\mathcal{C}_0$ which creates nodes x_0, x_1, y_1, z_0, and z_∞. (See Fig. 1.)

Step (2): If $\gamma > 0$, then add components $\mathcal{C}_1(1), \ldots, \mathcal{C}_1(\gamma)$, creating nodes $x_{\alpha_1}, \ldots x_{\alpha_\gamma}$. Each of these components is adjoined to the network at the nodes x_0, x_1, z_∞, which have already been created at Step (1). (See Fig. 2 and Table I.)

Step (3): Repeatedly add components $\mathcal{C}_2, \ldots, \mathcal{C}_7$ to create nodes $x_{p_1(\alpha_1,\ldots,\alpha_\gamma)}, \ldots, x_{p_\beta(\alpha_1,\ldots,\alpha_\gamma)}$. Steps (3a)–(3d) describe the creation of $x_{p_1(\alpha_1,\ldots,\alpha_\gamma)}, \ldots, x_{p_\beta(\alpha_1,\ldots,\alpha_\gamma)}$ as well as many intermediate nodes. (See Fig. 2 and Table I.)

- ***Step (3a)***: For any positive integer n, to create a node labeled x_n: First, add component $\mathcal{C}_4(1)$ to create node u_1. Then, for $i = 1, \ldots, n-1$, add component $\mathcal{C}_2(i)$ to create node z_i and add component $\mathcal{C}_6(i, 1)$ to create node x_{i+1}. This is possible since x_1, z_0, z_∞ have already been created.
- ***Step (3b)***: For any positive integer n, to create a node labeled x_{-n}: First, add component $\mathcal{C}_2(1)$ to create node z_1, and add component $\mathcal{C}_5(1)$ to create node u_{-1}. Then, for $i = 0, \ldots, n-1$, add component $\mathcal{C}_6(-i, -1)$ to create node x_{-i-1} and add component $\mathcal{C}_2(-i-1)$ to create node z_{-i-1}.
- ***Step (3c)***: For any positive integer n and any $\alpha \in \{\alpha_1, \ldots, \alpha_\gamma\}$ to create a node labeled x_{α^n}: First, add component $\mathcal{C}_3(\alpha)$ to create node y_α. Then, for $j = 1, \ldots, n-1$, add component $\mathcal{C}_2(\alpha^j)$ to create node z_{α^j} and add component $\mathcal{C}_7(\alpha^j, \alpha)$ to create node $x_{\alpha^{j+1}}$.
- ***Step (3d)***: To create nodes labeled by an arbitrary polynomial in $\boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$: Add various instances of components $\mathcal{C}_6$ and $\mathcal{C}_7$ to create nodes labeled by sums and products of labels of existing nodes created above. (Some instances of components $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$ may also have to be added in order to use $\mathcal{C}_6$ and $\mathcal{C}_7$.)

Step (4): Force each of the nodes $x_{p_1(\alpha_1, \ldots, \alpha_\gamma)}, \ldots, x_{p_\beta(\alpha_1,\ldots,\alpha_\gamma)}$ to demand message a.

To construct nodes labeled by arbitrary polynomials in $\boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$ in Step (3) of the algorithm, one can use Step (3a) to create all positive integer coefficients of the polynomials, use Step (3b) to create all negative integer coefficients of the polynomials, use Step (3c) to create all variable powers occurring in the polynomials, and finally use Step (3d) to combine the existing network nodes to create the desired polynomials.

This algorithm converts a single instance of the polynomial solvability problem to a single instance of the network scalar-linear solvability problem with the same answer. The procedure above is not the most efficient method to create the network $\mathcal{N}$ from the polynomial collection $\mathcal{P}$. A smaller network can in general be constructed. For example, to more efficiently create a node labeled x_n, where n is a positive integer, one could first create nodes $x_2, x_4, x_8, \ldots, x_{2^{\lfloor \log_2 n \rfloor}}$ and then use $\mathcal{C}_6$ on a subset of these nodes (corresponding to the binary expansion of n) to create x_n. This will add $O(\log n)$ nodes to obtain x_n, rather than $O(n)$ nodes as specified above. The same idea works for creating $x_{\alpha_i^n}$ as well. With these improvements, the construction produces a network whose size is linear in the size of the representation of the polynomial collection.

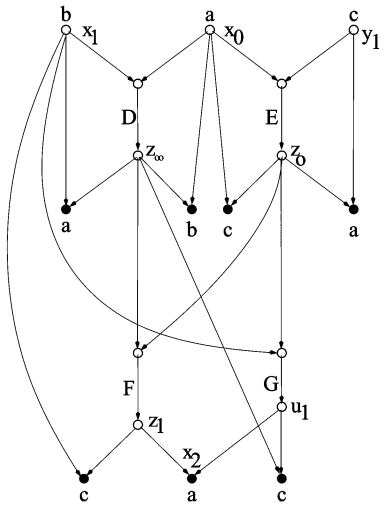

Fig. 3. A network constructed from the polynomial $p(\alpha) = 2$. All edges point downward. The top three nodes are sources generating messages b, a, and c from left to right, respectively. The seven solid nodes are receivers, each demanding one of the messages, as indicated below the receiver node.

III. A Network Construction Example for $\mathcal{P} = \{2\}$

In this section, we give a concrete example of constructing a network from a polynomial collection, specifically, a collection consisting of exactly one polynomial.

We illustrate here the construction of a network which has a scalar-linear solution over finite field F if and only if the constant polynomial $p(\alpha) = 2$ is solvable over F. Since $2 = 0$ if and only if F has even characteristic, the constructed network has a scalar-linear solution precisely for fields of order 2^m for all positive integers m.

We construct the network by adding, in the order listed, the following components: $\mathcal{C}_0$ (creating nodes x_0, x_1, y_1, z_0, z_∞), $\mathcal{C}_2(1)$ (creating node z_1), $\mathcal{C}_4(1)$ (creating node u_1), and $\mathcal{C}_6(1, 1)$ (creating node x_2). We then force node x_2 to be a receiver demanding message a (as a result, we can eliminate the out-edges from z_∞ and x_2 in $\mathcal{C}_6(1, 1)$). The resulting network is shown in Fig. 3 after some simplifications. We note that with a few minor alterations, the network in Fig. 3 can be redrawn to give us the Fano network, which was shown in [7] to be solvable precisely on power-of-two alphabet sizes and was constructed from the Fano matroid in [8].

One could also construct a network that has scalar-linear solutions precisely when the finite field F has odd characteristic, by using the polynomial collection $\mathcal{P} = \{2\alpha - 1\}$. One way to construct such a network is to add the following components in the order they are listed:

- $\mathcal{C}_0$ (creating nodes x_0, x_1, y_1, z_0, z_∞);
- $\mathcal{C}_2(1)$ (creating node z_1);
- $\mathcal{C}_4(1)$ (creating node u_1);
- $\mathcal{C}_1(1)$ (creating node x_{α_1});
- $\mathcal{C}_2(\alpha_1)$ (creating node z_{α_1});
- $\mathcal{C}_4(\alpha_1)$ (creating node u_{α_1});
- $\mathcal{C}_6(\alpha_1, \alpha_1)$ (creating node $x_{2\alpha_1}$);

- $\mathcal{C}_2(2\alpha_1)$ (creating node $z_{2\alpha_1}$);
- $\mathcal{C}_2(1)$ (creating node z_1);
- $\mathcal{C}_5(1)$ (creating node u_{-1});
- $\mathcal{C}_6(2\alpha_1, -1)$ (creating node $x_{2\alpha_1-1}$).

Finally, node $x_{2\alpha_1-1}$ would be forced to be a receiver demanding message a. This would involve adding ten new pieces to the network after the initial piece.

It was pointed out in [15] that a system of polynomials can be obtained from a network, such that the polynomial system is solvably equivalent to the network. It is interesting to illustrate this idea for the network in Fig. 3. There is one equation for each out-edge emanating from a node with at least two inputs (i.e., four equations, corresponding to the edges D, E, F, and G) and one equation for each message demand in the network (i.e., seven equations, corresponding the the demands of a, b, c, a, c, a, c). For each equation, there is a new variable introduced for each input associated to the out-edge or demand. In Fig. 3, each of the seven demands and each of the four edges D, E, F, and G has two inputs, so there are 22 variables among the equations. Note that since the network is acyclic, no variable will ever get multiplied by itself. Thus, the terms in the resulting polynomials will be either -1 or else products of variables, so no powers above one of variables will occur.

For this example network, we get the following conditions:

$$\begin{aligned}
D &= \alpha_1 b + \alpha_2 a \\
E &= \alpha_3 a + \alpha_4 c \\
F &= \alpha_5 D + \alpha_6 E = (\alpha_2\alpha_5 + \alpha_3\alpha_6)a + \alpha_1\alpha_5 b + \alpha_4\alpha_6 c \\
G &= \alpha_7 b + \alpha_8 E = \alpha_3\alpha_8 a + \alpha_7 b + \alpha_4\alpha_8 c \\
a &= \alpha_9 b + \alpha_{10} D = \alpha_2\alpha_{10} a + (\alpha_9 + \alpha_1\alpha_{10})b \\
b &= \alpha_{11} D + \alpha_{12} a = (\alpha_{12} + \alpha_2\alpha_{11})a + \alpha_1\alpha_{11} b \\
c &= \alpha_{13} a + \alpha_{14} E = (\alpha_{13} + \alpha_3\alpha_{14})a + \alpha_4\alpha_{14} c \\
a &= \alpha_{15} E + \alpha_{16} c = \alpha_3\alpha_{15} a + (\alpha_{16} + \alpha_4\alpha_{15})c \\
c &= \alpha_{17} b + \alpha_{18} F = \alpha_{18}(\alpha_2\alpha_5 + \alpha_6\alpha_3)a \\
&\quad + (\alpha_{17} + \alpha_1\alpha_5\alpha_{18})b + \alpha_4\alpha_6\alpha_{18} c \\
a &= \alpha_{19} F + \alpha_{20} G \\
&= (\alpha_2\alpha_5\alpha_{19} + \alpha_3\alpha_6\alpha_{19} + \alpha_3\alpha_8\alpha_{20})a \\
&\quad + (\alpha_1\alpha_5\alpha_{19} + \alpha_7\alpha_{20})b \\
&\quad + (\alpha_4\alpha_6\alpha_{19} + \alpha_4\alpha_8\alpha_{20})c \\
c &= \alpha_{21} D + \alpha_{22} G \\
&= (\alpha_2\alpha_{21} + \alpha_3\alpha_8\alpha_{22})a \\
&\quad + (\alpha_1\alpha_{21} + \alpha_7\alpha_{22})b \\
&\quad + \alpha_4\alpha_8\alpha_{22} c.
\end{aligned}$$

Thus, we get the system of polynomial equations:

$$\begin{aligned}
0 &= \alpha_2\alpha_{10} - 1 && (1)\\
0 &= \alpha_9 + \alpha_1\alpha_{10} && (2)\\
0 &= \alpha_{12} + \alpha_2\alpha_{11} && (3)\\
0 &= \alpha_1\alpha_{11} - 1 && (4)\\
0 &= \alpha_{13} + \alpha_3\alpha_{14} && (5)\\
0 &= \alpha_4\alpha_{14} - 1 && (6)\\
0 &= \alpha_3\alpha_{15} - 1 && (7)\\
0 &= \alpha_{16} + \alpha_4\alpha_{15} && (8)\\
0 &= \alpha_{18}(\alpha_2\alpha_5 + \alpha_6\alpha_3) && (9)\\
0 &= \alpha_{17} + \alpha_1\alpha_5\alpha_{18} && (10)\\
0 &= \alpha_4\alpha_6\alpha_{18} - 1 && (11)\\
0 &= \alpha_2\alpha_5\alpha_{19} + \alpha_3\alpha_6\alpha_{19} + \alpha_3\alpha_8\alpha_{20} - 1 && (12)\\
0 &= \alpha_1\alpha_5\alpha_{19} + \alpha_7\alpha_{20} && (13)\\
0 &= \alpha_4(\alpha_6\alpha_{19} + \alpha_8\alpha_{20}) && (14)\\
0 &= \alpha_2\alpha_{21} + \alpha_3\alpha_8\alpha_{22} && (15)\\
0 &= \alpha_1\alpha_{21} + \alpha_7\alpha_{22} && (16)\\
0 &= \alpha_4\alpha_8\alpha_{22} - 1. && (17)
\end{aligned}$$

This rather complicated set of equations is solvably equivalent to the single constant polynomial $p(x) = 2$, since they are both solvably equivalent to the network in Fig. 3.

Let us now verify this equivalence directly. The variables α_1, α_2, α_3, α_4, α_6, α_8, α_{18} are invertible from (4), (1), (7), (11), (17). Therefore, $\alpha_2\alpha_5 + \alpha_6\alpha_3 = 0$ from (9), and so (12) reduces to $\alpha_3\alpha_8\alpha_{20} = 1$, implying that α_{20} is invertible. Thus, α_{19} is invertible from (14). Now we can solve for $\alpha_9, \ldots, \alpha_{18}$, α_{20}, α_{22} in terms of the remaining variables in (1)–(8), (10)–(12), (17). Substituting these variables into (9), (13)–(16) gives the equivalent set of equations

$$\begin{aligned}
0 &= \alpha_2\alpha_5 + \alpha_6\alpha_3 && (18)\\
0 &= \alpha_1\alpha_5\alpha_{19} + \alpha_7\alpha_3^{-1}\alpha_8^{-1} && (19)\\
0 &= \alpha_6\alpha_{19} + \alpha_3^{-1} && (20)\\
0 &= \alpha_2\alpha_{21} + \alpha_3\alpha_4^{-1} && (21)\\
0 &= \alpha_1\alpha_{21} + \alpha_7\alpha_4^{-1}\alpha_8^{-1}. && (22)
\end{aligned}$$

Now α_{21} is invertible from (21) and α_7 is invertible from (22). Combining (19) and (20), and then (21) and (22) gives

$$\begin{aligned}
\alpha_1\alpha_5\alpha_8 &= \alpha_6\alpha_7 && (23)\\
\alpha_1\alpha_3\alpha_8 &= \alpha_2\alpha_7. && (24)
\end{aligned}$$

Multiplying (23) by α_2 and (24) by α_6, and then adding the resulting equations together gives $0 = 2\alpha_2\alpha_6\alpha_7$. So we must have $2 = 0$, which implies the field size is even. Conversely, if $2 = 0$, then we get a solution to (1)–(17) by setting

$$\alpha_1 = \cdots = \alpha_{22} = 1.$$

IV. Matroidality of Constructed Networks

First we review the concepts of matroids, matroidal networks, and the finite projective plane, each of which will be used in what follows.

A *matroid* $\mathcal{M}$ (e.g., see [20]) is an ordered pair $(\mathcal{S}, \mathcal{I})$, where $\mathcal{S}$ is a finite set and $\mathcal{I}$ is a set of subsets of $\mathcal{S}$ satisfying the following three conditions:

(11) $\emptyset \in \mathcal{I}$.

(12) If $I \in \mathcal{I}$ and $J \subset I$, then $J \in \mathcal{I}$.

(13) If $I, J \in \mathcal{I}$ and $|J| < |I|$, then $\exists\, e \in I - J$ such that $J \cup \{e\} \in \mathcal{I}$.

The set $\mathcal{S}$ is called the *ground set*, the members of $\mathcal{I}$ are called *independent sets*, and any subset of $\mathcal{S}$ not in $\mathcal{I}$ is called a *dependent set*. For any matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ and any $X \subset \mathcal{S}$, let $\mathcal{I}|X = \{I \subset X : I \in \mathcal{I}\}$, and let $\mathcal{M}|X = (X, \mathcal{I}|X)$.

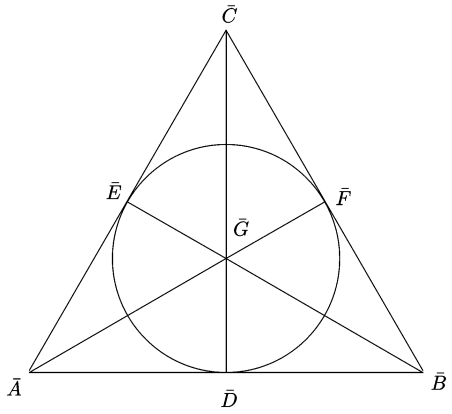

Fig. 4. Graphical depiction of the Fano matroid. The ground set is $\{\bar{A}, \bar{B}, \bar{C}, \bar{D}, \bar{E}, \bar{F}, \bar{G}\}$.

Then $\mathcal{M}|X$ is a matroid and the *rank* of X, denoted $\rho(X)$, is the (unique) size of a maximal independent set of $\mathcal{M}|X$. The rank of the matroid $\mathcal{M}$ is defined to be $\rho(\mathcal{S})$.

Let $\mathcal{N}$ be a network with message set μ, node set ν, and edge set ϵ. Let $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ be a matroid with rank function ρ. The network $\mathcal{N}$ is a *matroidal network* (see [8]) associated with $\mathcal{M}$ if there exists a function $f : \mu \cup \epsilon \rightarrow \mathcal{S}$ such that the following conditions are satisfied.

(M1) f is one-to-one on μ.

(M2) $f(\mu) \in \mathcal{I}$.

(M3) $\rho(f(\text{In}(x))) = \rho(f(\text{In}(x) \cup \text{Out}(x))), \forall\, x \in \nu$.

Example: The Fano matroid is depicted in Fig. 4. Each vertex represents a ground set element and any collection of vertices is an independent set if and only if it has at most three elements and is not one of the seven "lines" (i.e., $\bar{A}\bar{E}\bar{C}$, $\bar{A}\bar{D}\bar{B}$, $\bar{C}\bar{F}\bar{B}$, $\bar{E}\bar{G}\bar{B}$, $\bar{A}\bar{G}\bar{F}$, $\bar{C}\bar{G}\bar{D}$, and the circle $\bar{E}\bar{D}\bar{F}$). The network in Fig. 3 is matroidal, associated with the Fano matroid, as witnessed by the function f that satisfies

$$f(a) = \bar{A},\ f(b) = \bar{B},\ f(c) = \bar{C},\ f(D) = \bar{D}$$
$$f(E) = \bar{E},\ f(F) = \bar{F},\ f(G) = \bar{G}$$

and for every network node with only one input (i.e., edge or message) i and out-edges $o_1, \ldots, o_k$, we have

$$f(o_1) = \cdots = f(o_k) = f(i).$$

It is straightforward to verify (M1)–(M3) for this mapping.

It was shown in [8] that many interesting networks are matroidal, including all networks that are scalar-linearly solvable over a finite field (e.g., solvable multicast networks). The matroid used is a vector space over the finite field (with dimension the number of messages); the function f maps the messages to elementary vectors (vectors which are all 0 except for a single 1) and maps the edges to the corresponding "global coding vectors" (see, e.g., [14]) for the given scalar-linear code. (So this f is quite different from the local coding functions at each node.)

In [8], a method was presented for constructing, from given matroids, (matroidal) networks which reflect some of the matroids' properties. This construction was used to obtain networks used to prove various results in the literature [6], [7], [9]. For example, in [8], a network was constructed from the Vámos matroid that demonstrates the insufficiency of using Shannon-type information inequalities to compute network coding capacity. In what follows, we will prove that if a network is constructed from a solvable polynomial collection as in Section II, then the network is matroidal.

The network construction algorithm given in Section II was inspired by the 1936 work of Saunders MacLane in [17]. The proof of Theorem I.3 demonstrates the connection between the network construction, finite projective planes, and matroids.

For any positive integer n, a *projective plane* (e.g., see [3]) comprises a set of points, a set of lines, and an incidence relation between points and lines satisfying the following axioms.

(P1) Any two points are incident to exactly one line.

(P2) Any two lines are incident to exactly one point.

(P3) There exist four points, no three of which are incident to the same line.

One can show that, for any finite projective plane, there is a number n such that the plane has exactly n^2+n+1 points and n^2+n+1 lines, each line contains $n+1$ points, and each point lies on $n+1$ lines; n is called the *order* of the projective plane. A well-known (and presently unresolved) conjecture states that the order of any finite projective plane is a power of a prime.

Every finite projective plane induces a rank-three matroid as follows. Let $\mathcal{S}$ be the set of all points in the projective plane, let $\mathcal{I}$ be the collection of subsets of $\mathcal{S}$ of cardinality at most 3 that do not contain three collinear points, and let $\mathcal{M} = (\mathcal{S}, \mathcal{I})$. It is easy to see that $\mathcal{M}$ satisfies (I1) and (I2). Suppose $I, J \in \mathcal{I}$ where $|I| > |J|$. Then $|J| \in \{0, 1, 2\}$. If $|J| < 2$, then for any $v \in I - J$, we trivially have $J \cup \{v\} \in \mathcal{I}$. If $|J| = 2$ and if for each $v \in I - J$ we have $J \cup \{v\} \notin \mathcal{I}$, then the three points in I are collinear, contradicting $I \in \mathcal{I}$. Thus, $\mathcal{M}$ also satisfies (I3), and therefore $\mathcal{M}$ is a rank-3 matroid.

For any field F, one can construct a projective plane Π^F (of order $|F|$ if F is finite) as follows. Let

$$\Pi^F = (F \times F) \cup F \cup \{\infty\}$$

where two points (a, b) and (c, d) in $F \times F$ are said to have *slope* $s \in F$ if $a \neq c$ and

$$s = (d - b)(c - a)^{-1},$$

and slope $s = \infty$ if $a = c$. A *line* in Π^F consists of an element s of $F \cup \{\infty\}$ (called a *point at infinity*) together with a maximal set of points in $F \times F$ such that every two of them have slope $-1/s$ (where we make the convention that $v/0 = \infty$ and $v/\infty = 0$, for all nonzero $v \in F$). The set of all points at infinity is also considered a line. It can be verified that axioms (P1)–(P3) hold for Π^F.

In the case where F is the field of real numbers, one can depict the projective plane Π^F as the usual Euclidean plane together with the points at infinity, each of which represents the "intersection point" of a set of parallel lines; see Fig. 5. Many results that are intuitively clear from these depictions will carry over to the case where F is a finite field. (The geometric intuition

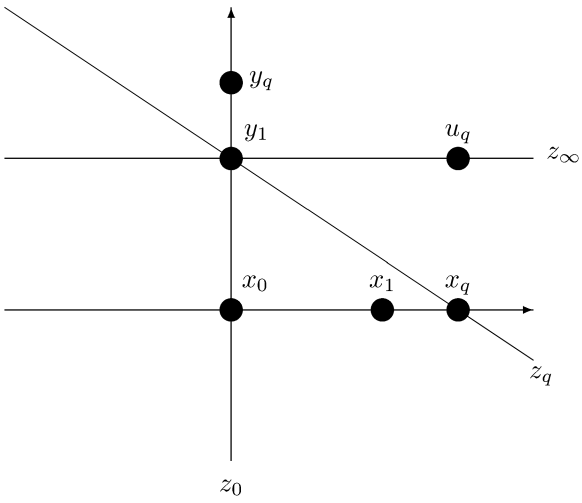

Fig. 5. The points x_q, y_q, z_q, and u_q in the projective plane.

from these depictions is useful, but is not a necessary part of the proofs to follow; one can simply verify the given formulas.) Some particular points to be used in what follows are the point x_q at offset q on the x-axis, the point y_q at offset q on the y-axis, the point u_q at offset q on the line $y = 1$, and the point z_q which is the point at infinity for the line through x_q and y_1 (or any line parallel to it). In particular, z_0 is on all vertical lines. The point at infinity for horizontal lines is denoted z_∞.

Given some initial points in the projective plane, one can determine additional points by drawing lines through the known points and finding their intersections. For instance, suppose we are given points x_0, x_1, y_1, z_0, and z_∞, as well as some other point x_q on the x-axis. Then we can construct u_q as the intersection of two lines, the vertical line through x_q (i.e., the line through the points x_q and z_0) and the horizontal line through y_1 (i.e., the line through y_1 and z_∞). We can construct z_q as the intersection of the line through x_q and y_1 with the line at infinity (the line through z_0 and z_∞). We can get z_1 in the same way. Then we can get y_q by intersecting the line through x_q and z_1 with the line through x_0 and z_0.

Then we can use these auxiliary points to construct additional points on the x-axis or elsewhere. Three cases of this are shown in Figs. 6–8. Fig. 6 shows the construction of u_{-q} as the intersection of the horizontal line $y = 1$ (determined by points y_1 and z_∞) with the line through the origin parallel to the line through y_1 and x_q (i.e., the line through x_0 and z_q). Fig. 7 shows addition of subscripts; the line through u_r parallel to the line through y_1 and x_q (i.e., the line through u_r and z_q) meets the x-axis at point x_{q+r}. Fig. 8 shows multiplication of subscripts; the line through y_r parallel to the line through y_1 and x_q (i.e., the line through u_r and z_q) meets the x-axis at point x_{qr} (note the similar triangles).

These are the constructions that we are imitating in the network context using the components in Section II. The network construction in [8] uses two basic methods for enforcing in the network a matroid dependency: a direct network dependency where the network is drawn so that one of the dependent values is put on an out-edge from a node whose inputs are the other members of the dependency, or an extra receiver node where

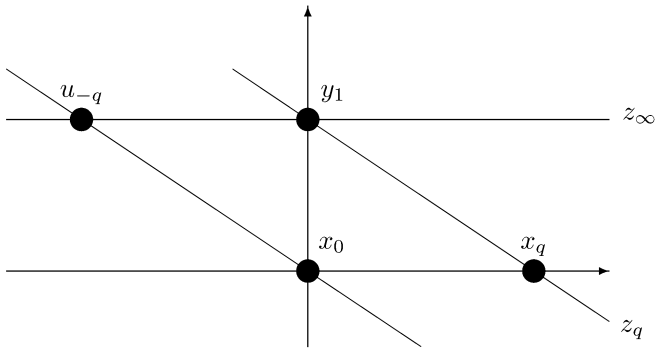

Fig. 6. Negation in the projective plane.

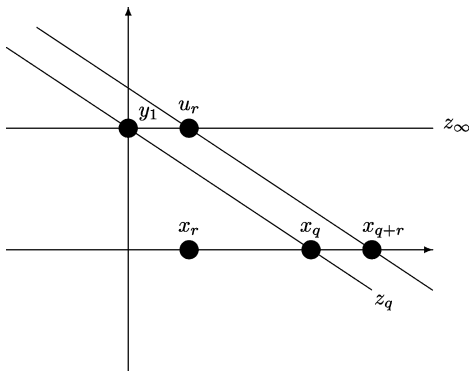

Fig. 7. Addition in the projective plane.

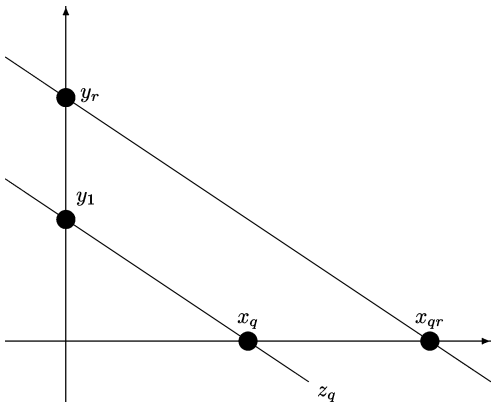

Fig. 8. Multiplication in the projective plane.

one member of the dependency (which must be a message) is demanded and the other members are inputs. Here the notion of dependency is collinearity in the projective plane. Given points p_1, p_2, p_3, p_4, if point p is to be the intersection of the line through p_1 and p_2 with the line through p_3 and p_4, then this can be modeled in the network by making p a direct descendant of p_1 and p_2 and putting in a demand for p_4 from p and p_3. The messages a, b, c of the network are assigned to points x_0, x_1, y_1, respectively; this gives enough flexibility that, for the cases we need, we can always arrange for p_4 to be one of these three points. This shows how the components $\mathcal{C}_0$ to $\mathcal{C}_7$ in Section II were constructed. (Components $\mathcal{C}_0$ and $\mathcal{C}_1$ construct points which are actually arbitrary points on only one known line, so the demand dependency line is the same as the direct dependency line.)

TABLE II
MAPPING UNDER f OF $\mathrm{In}(v) \cup \mathrm{Out}(v)$ FOR NODES v WITH IN-DEGREE TWO IN THE NETWORK $\mathcal{N}$ TO THE PROJECTIVE PLANE MATROID WITH GROUND SET Π^F, IN THE PROOF OF THEOREM I.3. IN THE FOURTH AND SEVENTH COLUMNS: IT CAN BE SEEN THAT THE THREE OUTPUT POINTS OF f ARE ALWAYS COLLINEAR IN THE PROJECTIVE PLANE Π^F, AND THUS HAVE RANK TWO IN THE MATROID

Component	Internal node		Mapping by f	Receiver node		Mapping by f
	Inputs	Output		Inputs	Output	
$\mathcal{C}_0$	y_1, x_0	z_0	$(0,1),(0,0),0$	y_1, z_0	a	$(0,1),0,(0,0)$
	x_0, x_1	z_∞	$(0,0),(1,0),\infty$	z_0, x_0	c	$0,(0,0),(0,1)$
				x_0, z_∞	b	$(0,0),\infty,(1,0)$
				z_∞, x_1	a	$\infty,(1,0),(0,0)$
$\mathcal{C}_1(i)$	x_0, x_1	x_{α_i}	$(0,0),(1,0),(\omega_i,0)$	x_{α_i}, z_∞	a	$(\omega_i,0),\infty,(0,0)$
$\mathcal{C}_2(q)$	z_0, z_∞	z_q	$0,\infty,\hat{q}$	z_q, x_q	c	$\hat{q},(\hat{q},0),(0,1)$
$\mathcal{C}_3(q)$	x_q, z_1	y_q	$(\hat{q},0),1,(0,\hat{q})$	y_q, z_0	a	$(0,\hat{q}),0,(0,0)$
$\mathcal{C}_4(q)$	x_q, z_0	u_q	$(\hat{q},0),0,(\hat{q},1)$	u_q, z_∞	c	$(\hat{q},1),\infty,(0,1)$
$\mathcal{C}_5(q)$	x_0, z_q	u_{-q}	$(0,0),\hat{q},(-\hat{q},1)$	u_{-q}, z_∞	c	$(-\hat{q},1),\infty,(0,1)$
$\mathcal{C}_6(q,r)$	z_q, u_r	x_{q+r}	$\hat{q},(\hat{r},1),(\hat{q}+\hat{r},0)$	x_{q+r}, z_∞	a	$(\hat{q}+\hat{r},0),\infty,(0,0)$
$\mathcal{C}_7(q,r)$	z_q, y_r	x_{qr}	$\hat{q},(0,\hat{r}),(\hat{q}\hat{r},0)$	x_{qr}, z_∞	a	$(\hat{q}\hat{r},0),\infty,(0,0)$

The demand of message a at each receiver node $x_{p_i(\alpha_1,\ldots,\alpha_\gamma)}$ corresponds to enforcing a two-point dependency between the points $(0,0)$ (i.e., from node x_0 which carries message a) and $(x_{p_i(\alpha_1,\ldots,\alpha_\gamma)},0)$.

As already noted, any network which is scalar-linearly solvable over a finite field is matroidal, so Theorem I.3 is a consequence of the proof of Theorem I.2 (to be given in the next section). However, we will give a direct proof here, because it completes the motivation for the construction in Section II.

Proof of Theorem I.3: Let

$$\mathcal{P} = \{p_1(\alpha_1,\ldots,\alpha_\gamma),\ldots,p_\beta(\alpha_1,\ldots,\alpha_\gamma)\}$$

be a polynomial collection and let $\mathcal{N}$ be a network constructed from $\mathcal{P}$ as in Section II. Let ν and ϵ, respectively, be the sets of nodes and edges of $\mathcal{N}$. Suppose F is a finite field such that $\mathcal{P}$ is solvable over F. Let $\omega_1,\ldots,\omega_\gamma \in F$ be such that

$$p_i(\omega_1,\ldots,\omega_\gamma) = 0$$

for all $i = 1,\ldots,\beta$.

Define a map g from the labeled nodes of $\mathcal{N}$ to Π^F as follows. For each polynomial $q \in \boldsymbol{Z}[\alpha_1,\ldots,\alpha_\gamma]$, let $\hat{q} = q(\omega_1,\ldots,\omega_\gamma)$ and let

$$\begin{aligned} g(x_q) &= (\hat{q},0) \\ g(y_q) &= (0,\hat{q}) \\ g(z_q) &= \hat{q} \\ g(u_q) &= (\hat{q},1) \\ g(z_\infty) &= \infty \end{aligned}$$

whenever these nodes appear in $\mathcal{N}$. By construction, every edge in $\mathcal{N}$ is either an in-edge or out-edge of some node x_q, y_q, z_q, and u_q. Now define a map

$$f : \mu \cup \epsilon \to \Pi^F$$

such that for each node $v \in \{x_q, y_q, z_q, u_q\}$, if e is an in-edge or out-edge of v, then $f(e) = g(v)$, and if m is a message generated by v, then $f(m) = g(v)$.

As noted earlier, the projective plane's set of points Π^F form the ground set of a rank-three matroid, whose independent sets $\mathcal{I}$ are the collections of three or fewer noncollinear points. Thus, to show $\mathcal{N}$ is matroidal, it suffices to verify that axioms (M1)–(M3) hold for the function f.

Since

$$\begin{aligned} f(a) &= g(x_0) = (0,0) \\ f(b) &= g(x_1) = (1,0) \\ f(c) &= g(y_1) = (0,1) \end{aligned}$$

we see that f is one-to-one on μ, so (M1) is satisfied. Also $(0,0)$, $(1,0)$, and $(0,1)$ do not lie on a line so they form an independent set in $\mathcal{M}$, and therefore (M2) is satisfied. Now, let v be an arbitrary node in $\mathcal{N}$. If v has in-degree zero or one, then $f(\mathrm{Out}(v)) = f(\mathrm{In}(v))$, so trivially $\rho(f(\mathrm{In}(v))) = \rho(f(\mathrm{In}(v) \cup \mathrm{Out}(v)))$.

We next examine in Table II every case where v has in-degree two (the maximum in-degree of any node in $\mathcal{N}$). In each case, it can be seen that f maps the two inputs of v to distinct points in Π^F implying that $\rho(f(\mathrm{In}(v))) = 2$, and f maps the two inputs of v and the output of v to three collinear points in Π^F (i.e., a dependent set in the matroid) implying that $\rho(f(\mathrm{In}(v) \cup \mathrm{Out}(v))) = 2$. This thus establishes that (M3) holds.

Finally, suppose $i \in \{1,\ldots,\beta\}$ and let v be the node labeled x_{p_i}. Then v is a receiver with in-degree one that demands message a, i.e., $\mathrm{In}(v) = x_{p_i}$ and $\mathrm{Out}(v) = a$. We have

$$\begin{aligned} f(x_{p_i}) &= g(x_{p_i}) = (\hat{p_i},0) = (p_i(\omega_1,\ldots,\omega_\gamma),0) = (0,0) \\ f(a) &= g(x_0) = (0,0) \end{aligned}$$

so $f(\mathrm{In}(x_{p_i}) \cup \mathrm{Out}(x_{p_i})) = f(\mathrm{In}(x_{p_i})) = (0,0)$.

Thus, the function f satisfies axiom (M3), so the network $\mathcal{N}$ is matroidal. □

MacLane [17] (see also [23, pp. 18–21]) used this construction as follows. Let $\mathcal{P}$ be a polynomial collection and let K be a finite field such that $\mathcal{P}$ has a solution over K. Then MacLane constructs a matroid $\mathcal{M}$ that is representable over K and such that, for any finite field F, if $\mathcal{M}$ is representable over F, then $\mathcal{P}$ has a solution over F. However, it is not necessarily true that, if $\mathcal{P}$ has a solution over F, then $\mathcal{M}$ is representable over F. Such an if-and-only-if result is not attainable in general for matroids; for instance, it is known that, if a matroid is representable over the two-element field and the three-element field, then it is representable over all finite fields [20, Theorem 6.6.3]. The extra flexibility of networks allows us to construct a network solvably equivalent to any given polynomial collection.

TABLE III
EDGE FUNCTIONS AND DECODING FUNCTIONS FOR THE COMPONENTS OF NETWORK $\mathcal{N}$. THE ARGUMENT OF $\mathcal{C}_1$ IS $i \in \{1,\ldots,\gamma\}$. THE ARGUMENTS OF $\mathcal{C}_2,\ldots,\mathcal{C}_7$ ARE ARBITRARY POLYNOMIALS $q, r \in \boldsymbol{Z}[\alpha_1,\ldots,\alpha_\gamma]$

Component	Edge function	Function of messages	Decoding functions
$\mathcal{C}_0$	$z_0 = x_0 - y_1$ $z_\infty = -x_0 + x_1$	$a - c$ $-a + b$	$a = y_1 + z_0$; $c = x_0 - z_0$ $b = x_0 + z_\infty$; $a = x_1 - z_\infty$
$\mathcal{C}_1(i)$	$x_{\alpha_i} = (1-\omega_i)x_0 + \omega_i x_1$	$(1-\omega_i)a + \omega_i b$	$a = x_{\alpha_i} - \omega_i z_\infty$
$\mathcal{C}_2(q)$	$z_q = z_0 + \hat{q} z_\infty$	$(1-\hat{q})a + \hat{q}b - c$	$c = x_q - z_q$
$\mathcal{C}_3(q)$	$y_q = x_q - \hat{q} z_1$	$(1-\hat{q})a + \hat{q}c$	$a = y_q + \hat{q} z_0$
$\mathcal{C}_4(q)$	$u_q = x_q - z_0$	$-\hat{q}a + \hat{q}b + c$	$c = u_q - \hat{q} z_\infty$
$\mathcal{C}_5(q)$	$u_{-q} = x_0 - z_q$	$\hat{q}a - \hat{q}b + c$	$c = u_{-q} + \hat{q} z_\infty$
$\mathcal{C}_6(q,r)$	$x_{q+r} = z_q + u_r$	$(1-\hat{q}-\hat{r})a + (\hat{q}+\hat{r})b$	$a = x_{q+r} - (\hat{q}+\hat{r}) z_\infty$
$\mathcal{C}_7(q,r)$	$x_{qr} = \hat{r} z_q + y_r$	$(1-\hat{q}\hat{r})a + \hat{q}\hat{r}b$	$a = x_{qr} - \hat{q}\hat{r} z_\infty$

V. NETWORK ALPHABETS AND POLYNOMIAL ROOTS

In this section, we prove our main result, namely, that the set of fields for which an arbitrary polynomial collection is solvable is identical to the set of fields for which a certain network (which depends on the polynomial collection) is scalar-linearly solvable. Thus, determining over which fields a general network has a scalar-linear solution is at least as difficult as determining over which fields a polynomial system is solvable.

Proof of Theorem I.2: Let

$$\mathcal{P} = \{p_1(\alpha_1,\ldots,\alpha_\gamma),\ldots,p_\beta(\alpha_1,\ldots,\alpha_\gamma)\}$$

be a polynomial collection with integer-valued coefficients and let $\mathcal{N}$ be a network constructed from $\mathcal{P}$, using the network construction algorithm in Section II. Let $\mathcal{N}$ have message set μ and edge set ϵ and let F be a field.

Part (1): Assume $\mathcal{P}$ is Solvable Over F: Suppose $\omega_1,\ldots,\omega_\gamma \in F$ are such that

$$p_i(\omega_1,\ldots,\omega_\gamma) = 0$$

for all $i = 1,\ldots,\beta$. For any polynomial $q \in \boldsymbol{Z}[\alpha_1,\ldots,\alpha_\gamma]$, let

$$\hat{q} = q(\omega_1,\ldots,\omega_\gamma) \in F.$$

We will describe a scalar-linear network code for $\mathcal{N}$ over F and prove it is a solution. The component $\mathcal{C}_0$ in $\mathcal{N}$ has two unspecified edge functions (at z_0 and z_∞) and an unspecified decoding function at each of four receiver nodes (to recover messages a, c, b, and a, respectively). Each occurrence of the components $\mathcal{C}_1,\ldots,\mathcal{C}_7$ has one unspecified edge function, and, at a receiver, one unspecified decoding function. Table III specifies these edge functions and decoding functions in the network code (these functions only have meaning for those polynomials corresponding to components that are part of the constructed network $\mathcal{N}$). Each edge function output in the table is also written, in the third column, in terms of the network messages a, b, and c. This column can be expressed equivalently as: for any polynomial q, we have

$$x_q = (1-\hat{q})a + \hat{q}b \tag{25}$$
$$u_q = -\hat{q}a + \hat{q}b + c \tag{26}$$
$$y_q = (1-\hat{q})a + \hat{q}c \tag{27}$$
$$z_q = (1-\hat{q})a + \hat{q}b - c. \tag{28}$$

Also, $z_\infty = -a + b$. (These formulas correspond to the parts of the definition of g in the proof of Theorem I.3, if we assign messages a, b, and c to the points $(0,0)$, $(1,0)$, and $(0,1)$ in the projective plane and suitably interpret "points at infinity.")

These formulas are proved from the edge function formulas by induction on the construction of the network. The base cases are the source message formulas

$$x_0 = a$$
$$x_1 = b$$
$$y_1 = c.$$

We now give the induction argument for each component $\mathcal{C}_i$ added to $\mathcal{N}$.

$$\begin{aligned}
\mathcal{C}_0: \quad & z_0 = x_0 - y_1 = a - c \\
& z_\infty = -x_0 + x_1 = -a + b \\
\mathcal{C}_1(i): \quad & x_{\alpha_i} = (1-\omega_i)x_0 + \omega_i x_1 = (1-\omega_i)a + \omega_i b \\
\mathcal{C}_2(q): \quad & z_q = z_0 + \hat{q} z_\infty = a - c + \hat{q}(-a + b) \\
& \quad = (1-\hat{q})a + \hat{q}b - c \\
\mathcal{C}_3(q): \quad & y_q = x_q - \hat{q} z_1 = (1-\hat{q})a + \hat{q}b - \hat{q}(b - c) \\
& \quad = (1-\hat{q})a + \hat{q}c \\
\mathcal{C}_4(q): \quad & u_q = x_q - z_0 = (1-\hat{q})a + \hat{q}b - (a - c) \\
& \quad = -\hat{q}a + \hat{q}b + c \\
\mathcal{C}_5(q): \quad & u_{-q} = x_0 - z_q = a - ((1-\hat{q})a + \hat{q}b - c) \\
& \quad = \hat{q}a - \hat{q}b + c \\
\mathcal{C}_6(q,r): \quad & x_{q+r} = z_q + u_r \\
& \quad = (1-\hat{q})a + \hat{q}b - c - \hat{r}a + \hat{r}b + c \\
& \quad = (1-\hat{q}-\hat{r})a + (\hat{q}+\hat{r})b \\
\mathcal{C}_7(q,r): \quad & x_{qr} = \hat{r} z_q + y_r \\
& \quad = \hat{r}((1-\hat{q})a + \hat{q}b) + (1-\hat{r})a + \hat{r}c \\
& \quad = (1-\hat{q}\hat{r})a + \hat{q}\hat{r}b.
\end{aligned}$$

Finally, recall that Step 4 of the construction in Section II imposed an additional demand on node x_{p_i} for each polynomial $p_i \in \mathcal{P}$. Since $p_i(\omega_1,\ldots,\omega_\gamma) = 0$ (i.e., $\hat{p}_i = 0$), the receiver node x_{p_i} carries the symbol

$$x_{p_i} = (1-0)a + 0b = a$$

and thus its demand of message a is satisfied. In summary, Table III demonstrates a code, all of whose demands are met, so the code is a solution to the network.

Part (2): Assume $\mathcal{N}$ is Scalar-Linearly Solvable Over F: We will show that the collection $\mathcal{P}$ must be solvable over F. Fix a specific scalar-linear solution to $\mathcal{N}$ over F. We next consider the symbols carried by various nodes in this solution.

In network component $\mathcal{C}_0$, node z_∞ depends only on a and b and node z_0 depends only on a and c; also, receivers demanding

a and c have z_0 as input and receivers demanding b and a have z_∞ as input. Thus, there exist nonzero constants $\pi_1, \pi_2, \theta_1, \theta_2 \in F$ (independent of a, b, and c) such that

$$z_0 = \pi_1 a + \pi_2 c = \pi_1(a - \pi c) \tag{29}$$

$$z_\infty = \theta_1 a + \theta_2 b = \theta_1(a - \theta b) \tag{30}$$

where

$$\pi = -\pi_2/\pi_1$$
$$\theta = -\theta_2/\theta_1.$$

By examining network components $\mathcal{C}_0$ and $\mathcal{C}_1(i)$, we see that for each $i \leq \gamma$, there must exist constants $\phi_1^{(i)}, \phi_2^{(i)} \in F$ (independent of a, b, c) such that

$$x_{\alpha_i} = \phi_1^{(i)} a + \phi_2^{(i)} b. \tag{31}$$

In component $\mathcal{C}_1(i)$, the demand a must be decoded at the receiver from nodes z_∞ and x_{α_i}, so it follows from (30) and(31) that $\phi_1^{(i)}\theta_2 - \phi_2^{(i)}\theta_1 \neq 0$, or equivalently

$$\theta\phi_1^{(i)} + \phi_2^{(i)} \neq 0.$$

For each $i \leq \gamma$, let

$$\omega_i = \phi_2^{(i)} \left(\theta\phi_1^{(i)} + \phi_2^{(i)}\right)^{-1}.$$

Then (31) can be rewritten as

$$x_{\alpha_i} = \left(\phi_1^{(i)} + \phi_2^{(i)}\theta^{-1}\right)\left((1-\omega_i)a + \omega_i\theta b\right). \tag{32}$$

We now show by induction that for each component $\mathcal{C}_i$ added to $\mathcal{N}$ we have the following constraints.

(i) If $i = 0, 1, 6, 7$ and the new node is x_s, then $\exists\, k \in F$ such that

$$x_s = k\left((1-\hat{s})a + \hat{s}\theta b\right). \tag{33}$$

(ii) If $i = 0, 3$ and the new node is y_s, then $\exists\, k \in F$ such that

$$y_s = k\left((1-\hat{s})a + \hat{s}\pi c\right). \tag{34}$$

(iii) If $i = 0, 2$ and the new node is z_s, then $\exists\, k \in F$ such that

$$z_s = k\left((1-\hat{s})a + \hat{s}\theta b - \pi c\right). \tag{35}$$

(iv) If $i = 4, 5$ and the new node is u_s, then $\exists\, k \in F$ such that

$$u_s = k\left(-\hat{s}a + \hat{s}\theta b + \pi c\right). \tag{36}$$

(These formulas are just formulas (25)–(28) with some additional constants introduced.)

The base cases are as follows.

- After $\mathcal{C}_0$ is added: $x_0 = a$ and $x_1 = b$ are immediate (satisfying (33) with $k = 1$), $y_1 = c$ is immediate (satisfying (34) with $k = \pi^{-1}$), and z_0 follows from (29) (satisfying (35) with $k = \pi_1$).
- After $\mathcal{C}_1(i)$ is added (for $1 \leq i \leq \gamma$): x_{α_i} is given in (32) (satisfying (33) with $k = \phi_1^{(i)} + \phi_2^{(i)}\theta^{-1}$).

For the induction step, consider a newly added component $\mathcal{C}_j$ (where $2 \leq j \leq 7$) which creates a new node x_s, y_s, z_s, or u_s for some $s \in \boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$. We handle these cases in what follows.

- Suppose x_s was added to $\mathcal{N}$ in component $\mathcal{C}_6(q, r)$, where $s = q + r$, for some $q, r \in \boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$. Then, from the structure of $\mathcal{C}_6(q, r)$, we conclude that there exist constants $A, B, C, D, k, k' \in F$ such that

$$\begin{aligned} x_{q+r} &= Az_q + Bu_r \\ &= Ak\left((1-\hat{q})a + \hat{q}\theta b - \pi c\right) \\ &\quad + Bk'\left(-\hat{r}a + \hat{r}\theta b + \pi c\right) \quad \text{[from (35), (36)]} \end{aligned} \tag{37}$$

$$\begin{aligned} a &= Cx_{q+r} + Dz_\infty \\ &= Cx_{q+r} + D\theta_1(a - \theta b). \quad \text{[from (30)].} \end{aligned} \tag{38}$$

The coefficients multiplying message c must be equal on both sides of (38), so, using (37), we get $0 = C(-\pi Ak + \pi Bk')$ or, equivalently, $Ak = Bk'$ (since $C \neq 0$ by (38)). Then (37) becomes

$$\begin{aligned} x_{q+r} &= Ak\left((1-\hat{q}-\hat{r})a + (\hat{q}+\hat{r})\theta b\right) \\ \therefore x_s &= Ak\left((1-\hat{s})a + \hat{s}\theta b\right). \end{aligned}$$

Alternatively, suppose x_s was added to $\mathcal{N}$ in component $\mathcal{C}_7(q, r)$, where $s = qr$, for some $q, r \in \boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$. Then, from the structure of $\mathcal{C}_7(q, r)$, we conclude that there exist constants $A, B, C, D, k, k' \in F$ such that

$$\begin{aligned} x_{qr} &= Az_q + By_r \\ &= Ak\left((1-\hat{q})a + \hat{q}\theta b - \pi c\right) \\ &\quad + Bk'\left((1-\hat{r})a + \hat{r}\pi c\right) \quad \text{[from (34), (35)]} \end{aligned} \tag{39}$$

$$\begin{aligned} a &= Cx_{qr} + Dz_\infty \\ &= Cx_{qr} + D\theta_1(a - \theta b). \quad \text{[from (30)]} \end{aligned} \tag{40}$$

The coefficients multiplying message c must be equal on both sides of (40), so, using (39), we get $0 = C(-\pi Ak + \pi B\hat{r}k')$ or equivalently, $Ak = Bk'\hat{r}$ (since $C \neq 0$ by (40)). Then (39) becomes

$$\begin{aligned} x_{qr} &= Bk'\left((1-\hat{q}\hat{r})a + \hat{q}\hat{r}\theta b\right) \\ \therefore x_s &= Bk'\left((1-\hat{s})a + \hat{s}\theta b\right). \end{aligned}$$

Thus, (33) is established for $i = 6, 7$.

- Suppose the node y_s was added to $\mathcal{N}$ in component $\mathcal{C}_3(s)$ after x_s was added to $\mathcal{N}$. From the structure of $\mathcal{C}_3(s)$, we conclude that there exist constants $A, B, C, D, k, k' \in F$ such that

$$\begin{aligned} y_s &= Ax_s + Bz_1 \\ &= Ak\left((1-\hat{s})a + \hat{s}\theta b\right) \\ &\quad + Bk'\left(\theta b - \pi c\right) \quad \text{[from (33), (35)]} \end{aligned} \tag{41}$$

$$\begin{aligned} a &= Cy_s + Dz_0 \\ &= Cy_s + D\pi_1(a - \pi c). \quad \text{[from (29)]} \end{aligned} \tag{42}$$

The coefficients multiplying messages a, b, and c must be equal on both sides of (42), so, using (41), we get

$$1 = ACk(1-\hat{s}) + D\pi_1 \tag{43}$$

$$0 = ACk\hat{s}\theta + BCk'\theta \tag{44}$$

$$0 = -BCk'\pi - D\pi\pi_1. \tag{45}$$

Note that $C \neq 0$, for otherwise we would have $D = 0$ by (45) which would then contradict (43). Thus, (44) implies $Ak\hat{s} = -Bk'$. Then (41) becomes

$$y_s = Ak\left((1-\hat{s})a + \hat{s}\pi c\right).$$

Thus, (34) is established for $i = 3$.

- Suppose the node z_s was added to $\mathcal{N}$ in component $\mathcal{C}_2(s)$ after y_s was added to $\mathcal{N}$. From the structure of $\mathcal{C}_2(s)$, we conclude that there exist $A, B, C, D, k \in F$ such that

$$\begin{aligned} z_s &= Az_0 + Bz_\infty \\ &= A\pi_1(a - \pi c) + B\theta_1(a - \theta b) \quad \text{[from (30), (29)]} \end{aligned} \tag{46}$$

$$\begin{aligned} c &= Cz_s + Dx_s \\ &= Cz_s + Dk\left((1-\hat{s})a + \hat{s}\theta b\right). \quad \text{[from (33)].} \end{aligned} \tag{47}$$

The coefficients multiplying messages a, b, and c must be equal on both sides of (47), so, using (46), we get

$$0 = AC\pi_1 + BC\theta_1 + Dk(1-\hat{s}) \tag{48}$$

$$0 = -BC\theta_1\theta + Dk\theta\hat{s} \tag{49}$$

$$1 = -AC\pi\pi_1. \tag{50}$$

Adding $\hat{s}$ times (48) to $(\hat{s}-1)\theta^{-1}$ times (49) gives $0 = C(B\theta_1 + \hat{s}A\pi_1)$ or, equivalently, $\hat{s}A\pi_1 = -B\theta_1$ (since $C \neq 0$ by (50)). Then (46) becomes

$$z_s = A\pi_1((1-\hat{s})a + \hat{s}\theta b - \pi c).$$

Thus, (35) is established for $i = 2$.

- Suppose the node u_s was added to $\mathcal{N}$ in component $\mathcal{C}_4(s)$ after z_s was added to $\mathcal{N}$. From the structure of $\mathcal{C}_4(s)$, we conclude that there exist $A, B, C, D, k \in F$ such that

$$\begin{aligned} u_s &= Ax_s + Bz_0 \\ &= Ak\left((1-\hat{s})a + \hat{s}\theta b\right) \\ &\quad + B\pi_1(a - \pi c) \qquad \text{[from (29), (33)]} \end{aligned} \tag{51}$$

$$\begin{aligned} c &= Cu_s + Dz_\infty \\ &= Cu_s + D\theta_1(a - \theta b). \quad \text{[from (30)]} \end{aligned} \tag{52}$$

The coefficients multiplying messages a, b, and c must be equal on both sides of (52), so, using (51), we get

$$0 = (1-\hat{s})ACk + \pi_1 BC + \theta_1 D \tag{53}$$

$$0 = \hat{s}\theta ACk - \theta_1\theta D \tag{54}$$

$$1 = -BC\pi_1\pi. \tag{55}$$

If we multiply (53) by θ and add the result to (54), then we get $0 = \theta C(Ak + \pi_1 B)$, or equivalently (since $C \neq 0$ by (55)), $Ak = -B\pi_1$. Then (51) becomes

$$u_s = Ak\left(-\hat{s}a + \hat{s}\theta b + \pi c\right).$$

Thus, (36) is established for $i = 4$.

Alternatively, suppose the node u_s was added to $\mathcal{N}$ in component $\mathcal{C}_5(-s)$. From the structure of $\mathcal{C}_5(s)$, we conclude that there exists $A, B, C, D, k \in F$ such that

$$\begin{aligned} u_s &= Ax_0 + Bz_{-s} \\ &= Aa + Bk\left((1+\hat{s})a - \hat{s}\theta b - \pi c\right) \quad \text{[from } \mathcal{C}_0\text{, (35)]} \end{aligned} \tag{56}$$

$$\begin{aligned} c &= Cu_s + Dz_\infty \\ &= Cu_s + D\theta_1(a - \theta b). \qquad \text{[from (30)].} \end{aligned} \tag{57}$$

The coefficients multiplying messages a, b, and c must be equal on both sides of (57), so, using (56), we get

$$0 = AC + BCk(1+\hat{s}) + \theta_1 D \tag{58}$$

$$0 = -BCk\hat{s}\theta - \theta_1\theta D \tag{59}$$

$$1 = -BCk\pi. \tag{60}$$

If we multiply (58) by θ and add the result to (59), then we get $0 = C\theta(A + Bk)$, or, equivalently (since $C \neq 0$ by (60)), $A = -Bk$. Then (56) becomes

$$u_s = -Bk\left(-\hat{s}a + \hat{s}\theta b + \pi c\right).$$

Thus, (36) is established for $i = 5$.

This completes the induction argument establishing the validity of (33)–(36) for all polynomials corresponding to nodes in $\mathcal{N}$.

Now, for the final demands in $\mathcal{N}$ to be satisfied, each of the nodes x_{p_i} must be able to recover message a. We must therefore have that for each $i = 1, \ldots, \beta$ there exists a nonzero constant $k \in F$ such that $x_{p_i} = ka$, or equivalently (by (33))

$$k'\left((1-\hat{p}_i)a + \hat{p}_i\theta b\right) = ka. \tag{61}$$

Clearly, since $\theta \neq 0$, we must have $\hat{p}_i = 0$ for all i. That is, $p_i(\omega_1, \ldots, \omega_\gamma) = 0$ for all $i \leq \beta$. Thus, the polynomial collection $\mathcal{P}$ is solvable over F. □

$\boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$ is a Noetherian ring by the Hilbert Basis Theorem (e.g., see [2, Theorem 4.6]), so every ideal of $\boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$ is finitely generated. For every infinite set of polynomials in $\boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$, there exists a finite set of polynomials in $\boldsymbol{Z}[\alpha_1, \ldots, \alpha_\gamma]$ that generates the same ideal. This implies the two sets of polynomials are solvable over precisely the same set of fields. Thus, without loss of generality, we may restrict attention to finite polynomial collections when determining solvable equivalence with networks.

The following lemma was given in 2003 by Baines and Vámos.

Lemma V.1: [1, Theorem 1.1] If a set of prime numbers is the set of characteristics of some polynomial collection, then the set is finite or co-finite.

Proof of Theorem I.9: It follows immediately from Theorem I.2, Lemma V.1, and Examples (1a) and (1b) in Section VI. □

We note that our definition of a network is slightly more general than other definitions in the literature, as we allow source nodes to have in-edges (so the edge functions at these nodes have both messages and in-edges as inputs) and receiver nodes to have out-edges. But it is easy to convert a network using these features to an equivalent network that does not (by adding some new output-only source nodes and input-only receiver nodes), so the results here also hold for the more restrictive definition of network. Also, it is easy to see that the Koetter–Médard mapping from networks to polynomial systems works just as well for the more general definition of network.

Part (2) of the proof of Theorem I.2 actually gives a correspondence (not one-to-one) between solutions to the polynomial

collection $\mathcal{P}$ over a field F and scalar-linear solutions to the network $\mathcal{N}$ over F. Given a scalar-linear solution to $\mathcal{N}$, the proof shows how to extract a solution $(\omega_1, \ldots, \omega_\gamma)$ to $\mathcal{P}$; it also gives nonzero constants $\theta_1, \theta, \pi \in F$ and a list of nonzero constants k such that formulas (30) and (33)–(36) hold, and completely define the network solution. (One constant k is needed for each x_s, y_s, z_s, and u_s appearing in the network, except for x_0, x_1, and y_1 at the sources. Recall that we have been assuming that each single-input node simply copies its input to all of its outputs; if we drop this assumption, then there will be additional constants specifying what happens at single-input nodes.) On the other hand, given a solution $(\omega_1, \ldots, \omega_\gamma)$ to $\mathcal{P}$ over F, one can choose the nonzero constants $\theta_1, \theta, \pi \in F$ and the k's arbitrarily and then use formulas (30) and (33)–(36) to define a solution to $\mathcal{N}$. The number of network solutions corresponding to each polynomial solution is $(|F|-1)^{3+c}$, where c is the number of k's used. (Alternatively, one could consider two solutions to a network to be "equivalent" if it is possible to get from one to the other by certain simple transformations. One such transformation is to fix a nonzero constant $r \in F$ and change the code by multiplying all coefficients being applied to a particular source message by r, and then multiplying all coefficients of decoding functions for that message by r^{-1}. Another is to multiply all input coefficients for a particular edge by r and then multiply each coefficient of that edge where it is used as an input for something else by r^{-1}. Then the equivalence classes of solutions to the network $\mathcal{N}$ are in one-to-one correspondence with the solutions to the polynomial collection $\mathcal{P}$.)

VI. Examples of Polynomial Collections

In this section, we examine over which fields certain collections of polynomials are solvable.

All polynomials listed in this subsection are over a single indeterminate variable α (except in part (8), where the variables are $\alpha_1, \ldots, \alpha_{n+1}$). When only a single polynomial is given in $\mathcal{P}$, the polynomial is referred to as p. By Theorem I.2, for each polynomial collection given below, there is a directed acyclic network which is solvable over precisely those fields where the polynomial collection is solvable. For any field F, let $\mathrm{char}(F)$ denote the characteristic of F.

(1) $\mathcal{P} = \{b\alpha + c\}$ for some fixed $b, c \in \boldsymbol{Z}$.
If $\mathrm{char}(F) \mid b$, then $p(\alpha) = c$ in which case $p(\alpha) = 0$ is solvable if and only if $\mathrm{char}(F) \mid c$. If $\mathrm{char}(F) \nmid b$, then b is invertible in F, in which case $p(\alpha) = 0$ is solvable over F by choosing $\alpha = -cb^{-1}$. In summary, the polynomial is solvable over F if and only if either $\mathrm{char}(F) \mid c$ or $\mathrm{char}(F) \nmid b$.
Some special cases include:
  (a) $\mathcal{P} = \{q_1 q_2 \cdots q_m\}$, where q_i is prime for each $i = 1, \ldots, m$.
  Then $p = 0$ is solvable over field F if and only if $\mathrm{char}(F) \in \{q_1, q_2, \ldots, q_m\}$.
  (b) $\mathcal{P} = \{(q_1 q_2 \cdots q_m)\alpha - 1\}$, where q_i is prime for each $i = 1, \ldots, m$.
  Then $p = 0$ is solvable over field F if and only if $\mathrm{char}(F) \notin \{q_1, q_2, \ldots, q_m\}$.

(2) $\mathcal{P} = \{\alpha^2 - 2\}$.
Then $p = 0$ is solvable over field F if and only if 2 has a square root in F.

(3) $\mathcal{P} = \{\alpha^2 + \alpha + 1\}$
Then $p = 0$ is not solvable over the binary field GF(2), but is solvable over GF(4).

(4) $\mathcal{P} = \{\alpha^2 + 1\}$
Then $p = 0$ is not solvable over $\boldsymbol{R}$, but is solvable over $\boldsymbol{C}$, GF(2), and GF(q) for odd primes q if and only if $q = 1 \pmod 4$.

(5) If $\mathcal{P}_1$ and $\mathcal{P}_2$ are solvable exactly over field collections $\mathcal{F}_1$ and $\mathcal{F}_2$ and the variables in $\mathcal{P}_1$ are distinct from the variables in $\mathcal{P}_2$, then $\mathcal{P}_1 \cup \mathcal{P}_2$ is solvable exactly over the fields in $\mathcal{F}_1 \cap \mathcal{F}_2$.

(6) If $\mathcal{P}_1$ and $\mathcal{P}_2$ are solvable exactly over field collections $\mathcal{F}_1$ and $\mathcal{F}_2$, then $\{p_1 p_2 : p_1 \in \mathcal{P}_1, p_2 \in \mathcal{P}_2\}$ is solvable exactly over the fields in $\mathcal{F}_1 \cup \mathcal{F}_2$.

(7) $\mathcal{P} = \{q, f(\alpha)\}$, q is prime, $k \geq 1$, f is an irreducible polynomial of degree k over GF(q).
Then $\mathcal{P}$ is solvable over finite field F if and only if $|F| = q^i$ and $k|i$. Thus, using the previous example, for any primes $q_1, \ldots, q_m$ and positive integers $k_1, \ldots, k_m$, we can create a polynomial collection which is solvable over F if and only if $|F| = q_j^i$ for some i and j such that $k_j | i$ and $1 \leq j \leq m$.

(8) $\mathcal{P} = \left\{ -1 + \alpha_{n+1} \prod_{1 \leq i < j \leq n} (\alpha_i - \alpha_j) \right\}$.
Then $p = 0$ is solvable over field F if and only if $|F| \geq n$.

VII. Complexity Questions

In this section, we briefly discuss some complexity questions relating to network scalar-linear solvability and polynomial collection solvability. First, we state four decision problems and then we present the status of various questions concerning their decidability.

(i) **Problem** $N_0(F)$
Given a directed acyclic network $\mathcal{N}$ and field F.
Is $\mathcal{N}$ scalar-linearly solvable over F?

(ii) **Problem** N_1 (*network scalar-linear solvability*)
Given a directed acyclic network $\mathcal{N}$.
Does there exist a finite field F such that $\mathcal{N}$ is scalar-linearly solvable over F?

(iii) **Problem** N_2
Given a directed acyclic network $\mathcal{N}$.
Is $\mathcal{N}$ scalar-linearly solvable over all sufficiently large prime finite fields?

(iv) **Problem** $P_0(R_1, R_2)$
Given rings R_1 and R_2 (where $R_1 \subseteq R_2$ or there is a canonical homomorphism from R_1 to R_2) and polynomials $p_1, \ldots, p_\beta \in R_1[\alpha_1, \ldots, \alpha_\gamma]$.
Does there exist $\omega_1, \ldots, \omega_\gamma \in R_2$ such that $p_i(\omega_1, \ldots, \omega_\gamma) = 0$ for all $i = 1, \ldots, \beta$?

The following facts summarize some of the present knowledge about the decidability of the four problems above and their relationship to each other.

- For any finite field F, Problem $N_0(F)$ is decidable by exhaustively searching over all possible (finitely many) scalar-linear network codes.
- If F is a field, then Problem $P_0(\boldsymbol{Z}, F)$ denotes what we have termed in this paper as a polynomial collection being solvable over a field. A restatement of our Theorem I.2

is that for every polynomial collection there exists a network such that for every field F, the answer to Problem $P_0(\boldsymbol{Z}, F)$ is "yes" if and only if the answer to Problem $N_0(F)$ is "yes." In other words, the construction gives a map from polynomial collections to networks which reduces $P_0(\boldsymbol{Z}, F)$ to $N_0(F)$, uniformly for all fields F. Conversely, the construction of Koetter and Médard [15] gives a map from networks to polynomial collections which reduces $N_0(F)$ to $P_0(\boldsymbol{Z}, F)$, uniformly for all fields F. Both of these reductions are many-to-one with linear input space blowups. (Actually, the unmodified Koetter–Médard reduction can lead to polynomial equations of size exponential in the size of the given network, but one can avoid this by adding auxiliary variables to the polynomial system.)

- Problem $P_0(\boldsymbol{Z}, \boldsymbol{Z})$ is Hilbert's Tenth Problem (HTP) [4], given in 1900, which was proven in 1973 to be undecidable [18]. Various generalizations of HTP have been areas of active research over the last century. For example, it is known that $P_0(\boldsymbol{R}, \boldsymbol{R})$ is decidable and it is presently not known whether $P_0(\boldsymbol{Q}, \boldsymbol{Q})$ is decidable.
- Problem $N_0(\boldsymbol{Q})$ is decidable if and only if Problem $P_0(\boldsymbol{Q}, \boldsymbol{Q})$ is decidable. Thus, determining the decidability of scalar-linear solvability of networks over rational alphabets is of the same difficulty as the presently open question of the decidability of the HTP generalization to the rationals.
- If the answer to Problem $N_0(\boldsymbol{Q})$ is "yes," then the answer to Problem N_2 is "yes," since any scalar-linear network solution over $\boldsymbol{Q}$ is also a solution over sufficiently large prime finite fields. However, the converse need not be true. For example, the polynomial

$$p(\alpha_1, \alpha_2, \alpha_3, \alpha_4) = \alpha_1^2 + \alpha_2^2 + \alpha_3^2 + \alpha_4^2 + 1$$

cannot equal zero when the variables are restricted to $\boldsymbol{Q}$, but can equal zero over every prime field GF(q) since the integer $q - 1$ can always be written as a sum of four perfect squares [13]. Thus, if we construct a network from p as in Section II, then Problem N_2 will answer "yes," but Problem $N_0(\boldsymbol{Q})$ will answer "no."

- Problem N_1 is decidable. To obtain an algorithm for N_1, one can first generate polynomials from a network, as indicated in [15] and then use a Gröbner basis calculation (which decides $P_0(\boldsymbol{Q}, \bar{\boldsymbol{Q}})$, where $\bar{\boldsymbol{Q}}$ denotes the algebraic closure of the rationals). There are only finitely many primes which occur as a factor of a denominator in this calculation (call these the "exceptional primes"). If the calculation indicates that there is a solution to the polynomial collection over an algebraic extension of $\boldsymbol{Q}$, then for any nonexceptional prime q the same computation will show that there is a solution over a finite algebraic extension of GF(q) (i.e., over some finite field of characteristic q). If the calculation shows that there is no solution in any algebraic extension of $\boldsymbol{Q}$, then there will also be no solution in any finite field whose characteristic is a nonexceptional prime; for each exceptional prime q, we can then run a separate Gröbner basis calculation modulo q to determine whether there is a solution over a finite field of characteristic q. (See [1] for more details.)
- If the word "scalar" is changed to "vector" in Problem $N_0(F)$, then the decidability of the problem is no longer known (this assumes an unspecified vector dimension, since a fixed vector dimension would clearly yield a finite search).
- If the phrase "scalar-linearly" is omitted from Problem N_1 (and "field" is changed to "alphabet"), then it becomes the *network solvability* problem, whose decidability also remains unknown.

ACKNOWLEDGMENT

The authors wish to thank Sidharth Jaggi and Søren Riis for a helpful discussion. They also thank two anonymous reviewers for excellent comments and suggestions to improve the manuscript.

REFERENCES

[1] R. Baines and P. Vámos, "An algorithm to compute the set of characteristics of a system of polynomial equations over the integers," *J. Symb. Comput.*, vol. 35, pp. 269–279, 2003.
[2] T. Becker, V. Weispfenning, and H. Kredel, *Gröbner Bases: A Computational Approach to Commutative Algebra*. New York: Springer-Verlag, 1993.
[3] L. M. Blumenthal, *A Modern View of Geometry*. New York: Dover, 1961.
[4] M. Davis, "Hilbert's tenth problem is unsolvable," *Amer. Math. Monthly*, vol. 80, pp. 233–269, 1973.
[5] R. Dougherty, C. Freiling, and K. Zeger, "Linearity and solvability in multicast networks," *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2243–2256, Oct. 2004.
[6] R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of linear coding in network information flow," *IEEE Trans. Inf. Theory*, vol. 51, no. 8, pp. 2745–2759, Aug. 2005.
[7] R. Dougherty, C. Freiling, and K. Zeger, "Unachievability of network coding capacity," *IEEE Trans. Inf. Theory (Joint issue with IEEE/ACM Trans. Netw.)*, vol. 52, no. 6, pp. 2365–2372, Jun. 2006.
[8] R. Dougherty, C. Freiling, and K. Zeger, "Networks, matroids, and non-Shannon information inequalities," *IEEE Trans. Inf. Theory*, vol. 53, no. 6, pp. 1949–1969, Jun. 2007.
[9] R. Dougherty and K. Zeger, "Nonreversibility and equivalent constructions of multiple-unicast networks," *IEEE Trans. Inf. Theory*, vol. 52, no. 11, pp. 5067–5077, Nov. 2006.
[10] M. Feder, D. Ron, and A. Tavory, "Bounds on linear codes for network multicast," in *Proc. Electronic Colloquium on Computational Complexity (ECCC)*, 2003, pp. 1–9.
[11] C. Fragouli and E. Soljanin, *Network Coding Fundamentals*. Boston, MA: Now Publishers, 2007.
[12] T. Ho, D. Karger, M. Médard, and R. Koetter, "Network coding from a network flow perspective," in *Proc. IEEE Int. Symp. Information Theory*, Yokohama, Japan, Jun. 2003, p. 441.
[13] K. Ireland and M. Rosen, *A Classical Introduction to Modern Number Theory*. New York: Springer-Verlag, 1990.
[14] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 1973–1982, Jun. 2005.
[15] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2003.
[16] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.
[17] S. MacLane, "Some interpretations of abstract linear dependence in terms of projective geometry," *Amer. J. Math.*, vol. 58, no. 1, pp. 236–240, Jan. 1936.
[18] Y. V. Matiyasevich, *Hilbert's Tenth Problem*. Cambridge, MA: MIT Press, 1993.
[19] M. Médard, M. Effros, T. Ho, and D. Karger, "On coding for nonmulticast networks," in *Proc. 41st Annu. Allerton Conf. Communication Control and Computing*, Monticello, IL, Oct. 2003.
[20] J. G. Oxley, *Matroid Theory*. New York: Oxford Univ. Press, 1992.
[21] A. L. Rasala and E. Lehman, "Complexity classification of network information flow problems," in *Proc. 41st Annu. Allerton Conf. Communication Control and Computing*, Monticello, IL, Oct. 2003.
[22] S. Riis, "Linear versus nonlinear boolean functions in network flow," in *Proc. 38th Annu. Conf. Information Sciences and Systems (CISS)*, Princeton, NJ, Mar. 2004.
[23] N. L. White, Ed., *Combinatorial Geometries, Encyclopedia of Mathematics and Its Applications*. Cambridge, U.K.: Cambridge Univ. Press, 1987.
[24] R. W. Yeung, *A First Course in Information Theory*. Boston, MA: Kluwer, 2002.
[25] R. W. Yeung, R. S.-Y. Li, N. Cai, and Z. Zhang, *Network Coding Theory*. Boston, MA: Now Publishers, 2006.